数理解析研究所講究録
第1769巻 2011年 90-100

Time Complexity of Square Pattern Generation on Two Dimensional Cellular Automata

渡辺 識　　大川 知
Satoru Watanabe　　Satoshi Okawa
会津大学
The University of Aizu

1 Introduction

A dot matrix method is used to express the two dimensional patterns such as figures or characters in printers or display. In this method, as the patterns are obtained by dots on lattices of fixed size, several dot patterns for each pattern should be prepared depending on the screen size. In recent years, several methods to draw some patterns on two dimensional cellular automata are proposed[4][6]. The methods are independent from the size of screen which is ralized by a cellular automaton. By using Firing Squad Synchronization(FSS), Komatsu has proposed methods to draw a square of maximum size in the center of the screen which is an $m \times n$ cellular automaton with $\frac{5}{2}m + \frac{9}{2}n - 8$ steps[4]. Watanabe and Okawa have proposed another method to draw a square of maximum size in the center of the screen by using FSS and a method to draw a square of maximum size without using FSS[6].

In this paper, we review the definitions and methods to draw a pattern, and we evaluate functions for the time complexity. First, we define two dimensional patterns as equivalence classes which are obtained by the similarity relation defined by movings and scaling on two dimensional plane. For an $m \times n$ screen, we define a pattern generation as to display with appropriate size (and position) in the screen. Next, we discretize the screen, and we define a pattern generation on the discretized screen. Furthermore, establishing a correspondence between the discretized screen and cellular automata, we study the pattern generation on the cellular automata. In the last part, we show methods to draw a square and a diamond patterns on two dimensional cellular automata by using algorithms of gradual generation and instantaneous generation, and we evaluate functions for the time complexity of drawing the patterns.

2 Pattern Generation

Let $\mathbb{R}$ be a set of real numbers, and a two dimensional plane is denoted by $\mathbb{R} \times \mathbb{R}$. A set $F \subseteq \mathbb{R} \times \mathbb{R}$ is called a two dimensional figure, a set of all two dimensional figures is denoted by $\mathcal{F}$, that is $\mathcal{F} = \{F | F \subseteq \mathbb{R} \times \mathbb{R}\}$. A figure which is obtained with moving F by $d \in \mathbb{R} \times \mathbb{R}$ is denoted by $F + d = \{p + d | p \in F\}$, and a figure which is obtained with extending F by $a (a > 0)$ times is denoted by $a \cdot F = \{a \cdot p | p \in F\}$. We define mappings S_d and Z_a as follows respectively,

$$S_d(F) = F + d, \quad Z_a(F) = a \cdot F.$$

We define a similarity relation $\sim$ on $\mathcal{F}$ using S_d and Z_a as follows.

For $F_1, F_2 \in \mathcal{F}$,

$$F_1 \sim F_2 \Leftrightarrow F_2 = S_d Z_a(F_1) \ (= aF_1 + d).$$

The relation $\sim$ is an equivalence relation on two dimensional figures. We define a pattern as a equivalence class using this relation as follows.

Definition 1
For a figure F, a pattern $[F]$ containing F is defined by

$$[F] = \{F' | F' \sim F\},$$

and a set of pattern $\mathcal{P}$ is defined by

$$\mathcal{P} = \mathcal{F} / \sim = \{P | P = [F], F \in \mathcal{F}\}.$$

For any $m, n > 0$, $[0, m] \times [0, n] \subseteq \mathbb{R} \times \mathbb{R}$ is called a screen of size $m \times n$, and it denoted by $C_{m \times n}$, where $[a, b]$ is an interval $\{x | a \leq x \leq b\}$.

Definition 2
For a pattern $P \in \mathcal{P}$ assuming $P = [F]$, generation of P on $C_{m \times n}$ is to obtain a set $D \subseteq C_{m \times n}$ which satisfies following conditions.

1. $\exists a, d \quad D = S_d Z_a(F)$,
2. $\forall \epsilon > 0 \quad S_d Z_{a+\epsilon}(F) \not\subseteq C_{m \times n}$.

Following discussion, we assume that m and n are integers for simplicity. When we display a figure in a screen, the screen has to be discretized, so we discretize $C_{m \times n}$ by dividing the width by $m - 1$ and dividing the length by $n - 1$. In this process, for each lattice point p, a copy of small screen is set on it. The small screen at the leftmost and the bottom position of the discretized screen is $c_{0,0}$, and a screen which is positioned in the ith position from the left side of the array and jth position from the bottom of the array is described by $c_{i,j}$, that is $c_{i,j} = C_{[i-0.5, i+0.5] \times [j-0.5, j+0.5]}$. We define the screen $C_{m,n}$ which is obtained by discretizing $C_{m \times n}$ as follows,

$$C_{m,n} = \{c_{i,j} | 0 \leq i \leq m, 0 \leq j \leq n, \quad i, j \in N\}.$$

We define a pattern generation on the discretized screen as follows.

Definition 3
For a pattern $P = [F] \in \mathcal{P}$, generation of P on $C_{m,n}$ is to obtain the following set $D' \subseteq C_{m,n}$,

$$D' = \{c_{i,j} | c_{i,j} \cap D \neq \phi\}.$$

3 Implementation with Cellular Automata

Two dimensional cellular automata consist of copies of a finite automaton (cell) which are positioned such as lattices. Each cell changes its own state to the state which is determined according to its own state and the adjacent cells' states. We call the own and adjacent cells *neighbors*, the function to determine the next state according to neighbors' states is called a local mapping. Each cell is expressed by $a_{i,j}$, that is, a cell placed at the cross point of the ith row and the jth column from the leftmost lowest cell. The interval of updating state is called a *step*. Formally, a two dimensional cellular automaton $\mathcal{M}$ is defined as follows,

$$\mathcal{M} = (M, Q, \sigma, N),$$

where $M \subset Z \times Z$ is a coordinate set where cells exist (we assume M is connected. Z means

a set of integers), Q is a set of states, $\sigma : Q \times Q^{|N|-1} \rightarrow Q$ is a local mapping, N is a set of neighbors. In this paper, we investigate $m \times n$ cellular automata which consist of m arrays of n cells, and we assume N as Neumann neighborhood, namely consisting of the own, upper, lower, right and left cells. In an initial configuration of $\mathcal{M}$, $a_{0,0}$ is active, and the all other cells are in *quiescent*.

By regarding each cell $a_{i,j}$ as $c_{i,j}$ in the discretized screen $C_{m,n}$, the set M can be regarded as the discretized screen $C_{m,n}$, and then, an $m \times n$ cellular automaton can be denoted as follows,

$$\mathcal{M} = (C_{m,n}, Q, \sigma, N).$$

Therefore, we regard a problem to generate P on $C_{m,n}$ as a problem to generate P on a cellular automaton $\mathcal{M}$, that is, a problem to construct $\mathcal{M}$ which generates P. To construct such $\mathcal{M}$ is to provide σ which specifies $D' \subseteq C_{m,n}$ at a certain time starting from the initial configuration. Here, D' is specified by letting $a_{i,j}$ be in a special state s if $a_{i,j} \in D'$.

We define a time complexity of pattern generation by algorithm A as follows.

Definition 4
For pattern P, when P is obtained by algorithm A on $\mathcal{M}$, the time complexity $Time(A(P))$ is the number of steps to draw pattern P starting from the initial configuration on $\mathcal{M}$ by using algorithm A.

4 Firing Squad Synchronization

We use Firing Squad Synchronization in a method to draw a pattern. The Firing Squad Synchronization(FSS) Problem for one dimensional cellular array was proposed by J.Myhill. We explain the basic idea of FSS as follows. At time $t = 0$, a cell which is in the end of the array is in *general* state, and the other cells are in *quiescent* state. The cells at the end know that they are located at the end of the array, and the other cells don't know their own location in the array. The goal of this problem is to design a set of states and a local mapping that lead all cells to a special state called *firing*.

The solution uses the divide and conquer method that divides a cellular array of length n into two cellular arrays of length $n/2$. By repeating this process, we obtain n cellular arrays of length 1. The main point for this solution is to find the center of the array. The cell which is in general state sends two signals p and q which is $1/3$ speed of p respectively. When p arrive at the reverse end, the cell sends back a signal $\bar{p}$. Then two signals $\bar{p}$ and q meet at the center cell of the array. The center cell changes its state to general. By this method, we obtain $3n$ steps solution easily. Furthermore, the optimum solution requiring only $2n - 2$ steps can be obtained by sophisticated method[7].

Next, we explain about FSS for two dimensional cellular array. For the FSS of two dimensional $m \times n$ rectangle array, an optimum solution requiring only $m + n + max\{m, n\} - 3$ steps has proposed[1]. In this paper, we use a simple method for two dimensional $n \times n$ square array. We explain the method as follows. The two dimensional $n \times n$ square array is divided into n arrays which are L form as shown in Figure1. The cell $a_{0,0}$ which is the general starts FSS in its row and column that has length n. After 2 steps, the cell $a_{1,1}$ becomes a general and starts FSS in next L form array which consists of the row and the column that has length $n - 1$. The first FSS finishes with $2n - 2$ steps, and the next FSS finishes with $2n - 4$ steps. Therefore, these two FSSs finish at the same time. By repeating this method, the FSS of square array finishes with $2n - 2$ steps.

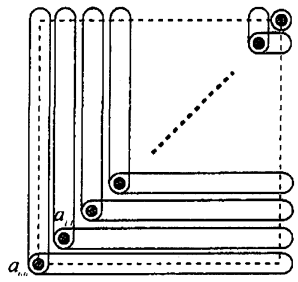

Figure 1: Firing Squad Synchronization in two dimensional square array

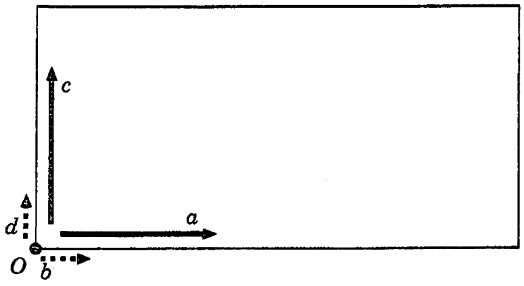

Figure 2: setting the center of the length and the width

5 Square Pattern Generation on Cellular automata

As examples of pattern generation, we show the drawings of patterns on an $m \times n$ cellular automaton.

First, we explain propagation of signals among cells in two dimensional cellular array. When a next cell of a cell in state s changes its own state to s at k steps, we call the signal specified by s propagates at speed $1/k$. A cell can send signals upper, lower, right, and left directions.

A cell can send a signal to the directions of diagonal 45 degrees. For example, a cell $a_{i,j}$ sends $1/1$ signals to $a_{i+1,j}$ and $a_{i,j+1}$ simultaneously, and then $a_{i+1,j}$ and $a_{i,j+1}$ send $1/1$ signal to $a_{i+1,j+1}$. As a result, $a_{i,j}$ sends a $1/2$ signal to $a_{i+1,j+1}$. We call the direction of the signal right upper. Cells can send signal to the other diagonal directions in the same way, we call the directions left upper, right lower, and left lower respectively.

Next, we show the drawings of a square and a diamond pattern on an $m \times n$ cellular automaton as follows. In the following examples, we assume that $m > n$.

5.1 square pattern

We investigate a method to generate the square of maximum size in the center of a given $m \times n$ cellular automaton. We call the algorithm for the method A_{s1}. We explain each process of the algorithm A_{s1} with a number of the steps of each process as follows.

(1)setting the center of the length and the width

The cell $a_{0,0}$ which is placed at O sends Signal a with speed $1/1$ and Signal b with speed $1/3$ to the right, and $a_{0,0}$ also sends Signal c with speed $1/1$ and Signal d with speed $1/3$ to the upper simultaneously, as shown in Figure 2.

After Signal a reached a cell in the end of the line(steps : $(\mathbf{m}-\mathbf{1})$), the cell sends back Signal $\bar{a}$ to the left with speed $1/1$, and then Signal $\bar{a}$ meets Signal b at H which is center of

the width. Furthermore, after Signal c reached a cell in the end of the line, the cell sends back Signal $\bar{c}$ to the lower with speed $1/1$, and then Signal $\bar{c}$ meets Signal d at V which is center of the length simultaneously, as shown in Figure 3. (steps : $(m-1)+(\mathbf{1/2m-1})$)

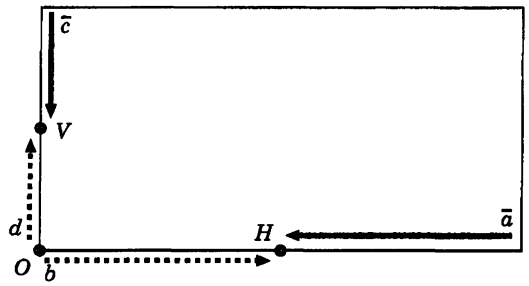

Figure 3: setting the center of the length and the width

(2) setting of the center of the square

Cells at H and V send Signal e and f to the upper and the right respectively. Cells which are passed by Signal e or f become a state to indicate a trace of Signal e or f. A signal hits the another signal's trace, and then we obtain the cell at T which is the center of the square, as shown in Figure 4. (steps :$(m-1)+(1/2m-1)+(\mathbf{1/2n-1})$)

The cell at H also sends Signal g and h to the left upper and right upper direction respectively to find sides of the square(Figure 4). Signal g and Signal h hit the trace of f, and then we obtain the cells at L and R which are centers of left and right side of the square, as shown in Figure 5. (steps :$(m-1)+(1/2m-1)+\mathbf{2(1/2n-1)}$)

(3) drawing the square

The cell at T sends Signal i, j, k, and l with speed $1/2$ to the corners of the square A, B, C, and D respectively, as shown in Figure 5.

The cells that are passed by the Signal i and j change their own state to the special state s, and send Signal p with speed $1/1$ to the lower direction. Similarly, the cell that are passed by the Signal k and l send Signal p with $1/1$ to the upper direction. The cells that are passed by Signal p change their own states to the special state s. The cells at L and R send Signal t and u with speed $1/1$ to the center of the square respectively as shown in Figure 6. The cells that are passed by the Signal t and u change their own state to the special state s, and send Signal q to the upper and lower direction respectively. The cells that are passed by Signal q change their own states to the special state s.

All cells in area $ABCD$ become state s, we can obtain the square. (steps : $(m-1)+(1/2m-1)+2(1/2n-1)+(\mathbf{1/2n-1})$)

By summing up the number of total steps for the algorithm A_{s1}, we obtain the time complexity as follows,

$$Time(A_{s1}(Square)) = \frac{3}{2}m + \frac{3}{2}n - 5$$

By the method mentioned above, a figure is drawn gradually. For an instantaneous appearance of the figure at a certain time, we can generate the square using the firing squad synchronization instead of the method after step (3). The following algorithm A_{s2} shows a generation by the firing squad synchronization technique. In the algorithm A_{s2}, same processes are performed in four areas of square : left upper, right upper, left lower, and right lower. In each area, FSS is performed for two dimensional $n \times n$ square array. Therefore, each

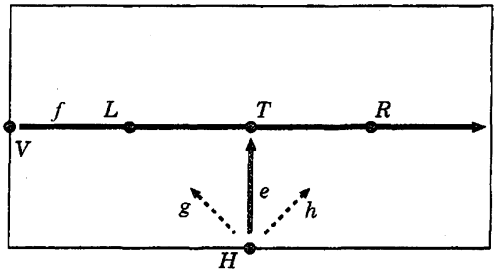

Figure 4: setting of the center and side of the square

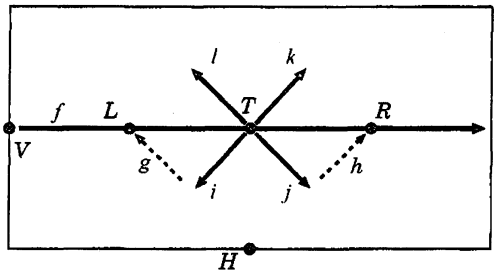

Figure 5: drawing the square

square area is divided into $n/2$ L form array. In the following explanation, we mention about only left upper area of the square for simplicity.

(3') starting of firing squad synchronization

The cell at T, which is center of the square, becomes a general of L form array which consists of column UT and row LT as shown in Figure 7, and FSS starts in the L form array. Furthermore, the cell at T sends Signal i with speed $1/2$ to the left upper for the FSS of the square area.

(4') drawing the square by the firing squad synchronization technique

After receiving the signal of FSS, the cell at L sends Signal j with speed $1/1$ to the upper to set an area of the square form, as shown in Figure 8. (steps :$(m-1)+(1/2m-1)+(1/2n-1)+(\mathbf{1/2n-1})$) Signal i and Signal j reach D at the same time, and then the FSS for the square area finishes. (steps :$(m-1)+(1/2m-1)+(1/2n-1)+(1/2n-1)+(\mathbf{1/2n-1})$)
The processes of (3') and (4') are performed in all four area, as shown in Figure 9, we can obtain the square.

By summing up the number of total steps for the algorithm A_{s2}, we obtain the time complexity as follows,

$$Time(A_{s2}(Square)) = \frac{3}{2}m + \frac{3}{2}n - 5$$

5.2 Diamond pattern

We investigate a method to generate a diamond pattern, which is a square that has corners in the center of top and bottom (right and left) sides of the given $m \times n$ cellular automaton. We call the algorithm for the method A_{d1}. We explain each process of the algorithm A_{d1} with a

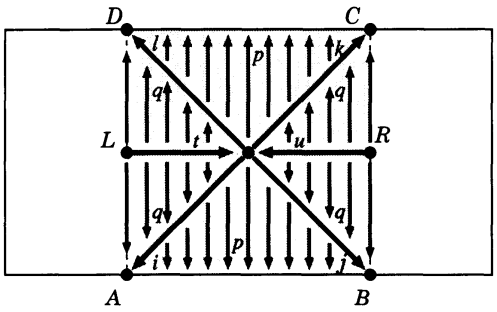

Figure 6: drawing the square

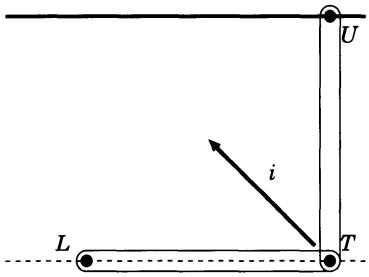

Figure 7: starting of firing squad synchronization

number of the steps of each process as follows.

(1) setting the centers and left and right corners of the diamond

We can obtain T which is the center of the diamond and L, R which are the left and right corners of the diamond by using the same manner for the previous square as shown in Figure 4.

(2) setting the top and bottom corners of the diamond

The cell at T sends Signal i and j to the upper and the lower direction respectively, and then we obtain a cell at U which is the top corner and a cell at H which is the bottom corner of the diamond respectively. At this time, we obtain cells at L and R which are left and right corners of the diamond simultaneously, as shown in Figure 10. (steps :$(m-1)+(1/2m-1)+(1/2n-1)+(\mathbf{1/2n-1})$)

(3) drawing the diamond

The cells at four corners H, U, L, and R send Signal u to the center of the diamond simultaneously as follows.

The cell at H sends Signal i and j to the left upper and the right upper direction with speed $1/2$ respectively. The cells that are passed by Signal i and j change their own state to the special state s, and send Signal u with speed $1/1$ to the upper direction.

The cell at U sends Signal k and l to the right lower and the left lower direction with speed $1/2$ respectively. The cells that are passed by Signal k and l change their own state to the special state s, and send Signal u with speed $1/1$ to the lower direction.

The cell at L sends Signal p and q to the right upper and the right lower direction with speed $1/2$ respectively. The cells that are passed by Signal p and q change their own state to

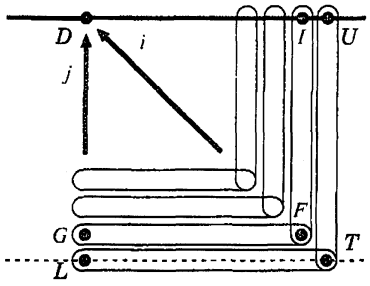

Figure 8: drawing the square by firing squad synchronization

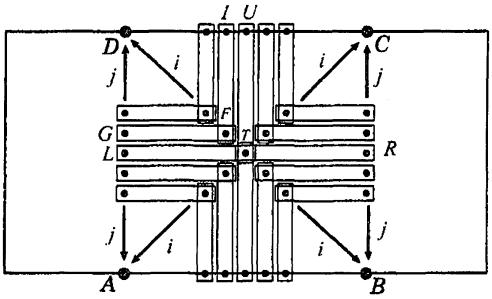

Figure 9: drawing the square by firing squad synchronization in four area

the special state s, and send Signal u with speed $1/1$ to the right direction.

The cell at R sends Signal r and t to the left lower and the left upper direction with speed $1/2$ respectively. The cells that are passed by Signal r and t change their own state to the special state s, and send Signal u with speed $1/1$ to the left direction.

The cells that are passed by Signal u change their own state to the special state s, and the signals of u hit each other and stop at those points as shown in Figure 9. All cells in area $HRUL$ become state s, we can obtain the square. (steps :$(m-1)+(1/2m-1)+(1/2n-1)+(1/2n-1)+(\mathbf{1/2n-1})$)

By summing up the number of total steps of the algorithm A_{d1} we obtain the time complexity as follows,

$$Time(A_{d1}(Square)) = \frac{3}{2}m + \frac{3}{2}n - 5$$

We can also generate the diamond by using firing squad synchronization instead of the previous method after process of (3). The following algorithm A_{d_2} shows the generation by firing squad synchronization technique, which is arranged for the diamond pattern. In the algorithm A_{d2}, same processes are performed in four right triangles areas : left upper, right upper, left lower, and right lower. In each area, FSS is performed for two dimensional right triangles area. Therefore, each right triangles area is divided into $n/4$ L form array. In the following explanation, we mention about only left upper area of the diamond for simplicity.

(3') starting of firing squad synchronization

The cell at T, which is center of the diamond, becomes a general of L form array which consists of column UT and row LT as shown in Figure 12, and FSS starts in the L form array. Furthermore, the cell at T sends Signal i with speed $1/4$ to the left upper direction for the FSS of the right triangles area.

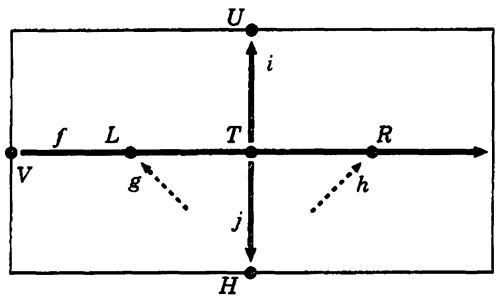

Figure 10: setting the top and bottom corners of the diamond

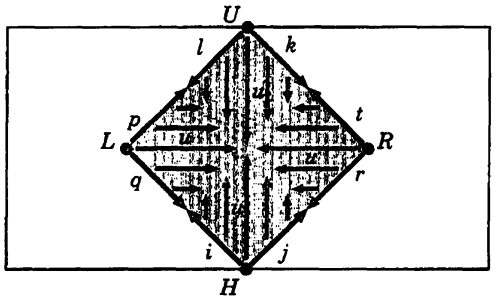

Figure 11: drawing the diamond

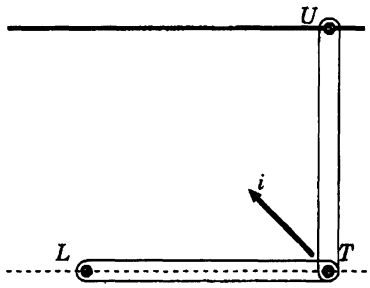

Figure 12: starting of firing squad synchronization

(4') drawing the diamond by the firing squad synchronization technique

After receiving a signal for FSS, the cell at L sends Signal j with speed $1/2$ to the right upper direction, and the cell at U sends Signal k with speed $1/2$ to the left lower direction as shown in Figure 13. We obtain cells at G and I, which are the end points of FSS, by Signal j and k. After receiving Signal i, the cell at F becomes general of FSS in next L form array which consists of the row GF and the column IF and the FSS starts in the L form array. Because the Signal i is $1/4$ signal, FSS starts every 4 steps. In the first column of length $n/2$ (LT and UT),the FSS needs $n-2$ steps, The FSS in the next column of length $n/2-2$ (GF and IF) needs $n-6$ steps. Therefore, these FSS finish at the same time. By Repeating this method, all FSS finish when the Signal i, j and k hit each other. (steps :$(m-1)+(1/2m-1)+(1/2n-1)+\mathbf{2(1/2n)-2}$)

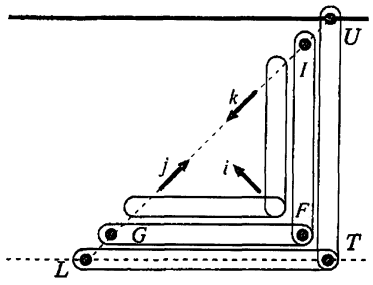

Figure 13: drawing the diamond by the firing squad synchronization technique

The process of (3') and (4') are performed in all four area, as shown in Figure 14, we can obtain the diamond.

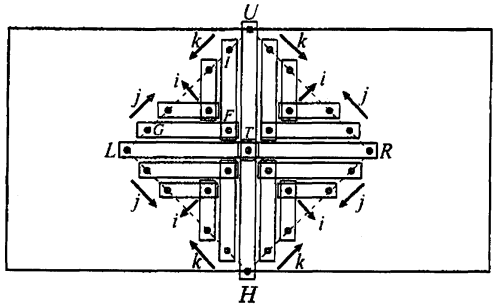

Figure 14: drawing the diamond by the firing squad synchronization technique

By summing up the number of total steps of the algorithm A_{d2} we obtain the time complexity as follows,

$$Time(A_{d2}(Square)) = \frac{3}{2}m + \frac{3}{2}n - 5$$

6 Conclusion

In this paper, we defined a pattern and a pattern generation. Next, we discretized the screen, and defined a pattern generation on the discretized screen. Furthermore, we studied a correspondence between the discretized screen and cellular automata, and we studied the pattern generation on the cellular automata. In the last part, we show methods to draw a square and a diamond patterns on two dimensional cellular automata, and we evaluate functons for the time complexities of drawing the patterns.

The optimal time complexities of drawing the square and diamond patterns are not known. We have studied to generate patterns which have complex form or characters, for example, patterns which consist of curves, or combinations of curves and straight lines. At present, we have to consider a method for pattern generation according to an individual figure. If any pattern can be generated using a standardized method, we can use it in a display such as an electric bulletin board or a printer.

Acknowledgements

The authors are grateful to the participants of the symposium "Algebras, Languages, Algorithms in Algebraic Systems and Computations"for their comments to improve the research

and its applications.

reference

[1] Y.Mizuno, A New Solution of the Firing Squad Synchronization Problem for Two Dimensional Rectangle Arrays, *University of Aizu, Grad.Thesis.*, March, 2005.

[2] M.Teraoka, et al. A Design of Generalized Optimum-Time Firing Squad Synchronization Algorithm for Two-Dimensional Cellular Arrays, *The 18th Annual Conf.of Japanese Society for Artificial Intelligence*, 3H1-04,2004.

[3] A.Settle and J.Simon, Smaller solutions for the firing squad, *Theoretical Computer Science*, 276, pp.83-109, 2002.

[4] T.Komatsu, Pattern Generation in Two Dimension Plane, *University of Aizu, Grad.Thesis.*, March, 2008.

[5] S.Wolfram, Two-Dimensional Cellular Automata, *Journal of Statistical Physics, vol 38, p901-946*, March 1985.

[6] S.Watanabe and S.Okawa, Pattern Generation on Two Dimensional Cellular Automata, *Forum on Information Technology 2009*, A-023.

[7] J. Mazoyer, A six state minimal time solution to the firing squad synchronization problem, *Theoretical Computer Science, vol.50, pp.183-238*,1987